Dialogs Are Dead, Long Live Dialogs: Building a Multimodal Automotive Application for Both Novices and Experts

Garrett Weinberg
Nuance Communications, Inc.
1 Wayside Rd.
Burlington, MA 01803, U.S.A.
garrett.weinberg@nuance.com

ABSTRACT
We describe a prototype car navigation system with a high level of sensitivity towards the differing abilities of users and the differing levels of visual and cognitive demand in real-world driving. The system, called VoiceCar, supports conventional, step-by-step, mixed-initiative voice dialogs and commands as well as more open-ended, entirely user-initiative interactions (which we term "anti-dialogs"). Much validation work remains to be done, but we hope that our demonstration system's support for both dialogs and anti-dialogs makes it more adaptable to a variety of driving situations, to users of all levels, and to ill-defined or shifting user intentions.

Categories and Subject Descriptors
H.5.2 [**User Interfaces**]: Graphical User Interfaces (GUI); Interaction Styles (e.g. commands, menus, forms, direct manipulation); Voice I/O

General Terms
Design, Human Factors

Keywords
Multimodality, dialogs, speech dialog systems

1. INTRODUCTION AND RELATED WORK
There is ample evidence that speech-based interaction with in-vehicle information systems (IVIS) improves such critical driving behaviors as lane keeping and reaction time as compared to manual interactions with the same systems (e.g., [1][3][5]). However less attention has been paid to multimodal (often speech + tactile) systems, or to driving situations that vary in cognitive or psychomotor demand levels over the course of a driving session. Castronovo et al. [2] found that both multimodal and speech-only control of HVAC functions resulted in less deviation from the ideal lane position than the use of a rotary knob alone, but their study was carried out using the highway Lane-Change Task [6], which is fairly uniform in nature (in terms of vehicle speed and maneuvers required).

Despite the findings cited above regarding voice interfaces' benefits to drivers, might there be certain road or traffic conditions in which manual input is more appropriate or even safer than voice input (especially considering the sometimes detrimental effects of recognition errors, when they do occur [4])?

Perhaps a mix of the two modalities is the most practical option. When one hand can come off the steering wheel and there is time for quick downward glances, users may prefer the simple efficiency of a touchscreen or knob. In heavy traffic or while actively maneuvering, however, reliance on the voice user interface (VUI) seems to make more sense.

It is this pragmatic, "best tool for the job" philosophy that has inspired the design for VoiceCar. We will introduce the system briefly in the following sections, but we hope the reader will also attend the demonstration of an early iteration of the system.

2. SYSTEM DETAILS
2.1 Voice and Multimodality
As one might guess from the prototype's name, speech is a first-class modality in VoiceCar rather than merely a supplement to tactile and visual interaction. All screens and functions of the application can be accessed by spoken commands or searches, and synthesized or recorded voice prompts announce all important changes of state. However, visible text and graphics play equally important roles (as will be further explained below), because users generally can comprehend the results of a search or command more quickly by reading than by listening.

2.2 Platform
VoiceCar is built on top of Nuance's production embedded application framework, meaning it can leverage all of the core automatic speech recognition (ASR) and text-to-speech (TTS) engines' advanced features (such as natural language understanding and multilingual acoustic models). Noise robustness and broad language and dialect coverage also come "for free" in this sense. Meanwhile, the use of the production framework means that VoiceCar's advancements in the application tier (e.g., in dialog and grammar design and in GUI/VUI integration) can more easily cross-pollinate customer projects. This means that although VoiceCar is primarily a research and sales tool, there is very little "throwaway" work.

The application framework is written in standard ANSI C, so we wanted a graphics toolkit with nearly as wide an array of porting options, but one which also offered a robust Model-View-

AutomotiveUI'11, November 30–December 2, 2011, Salzburg, Austria.
Adjunct Proceedings

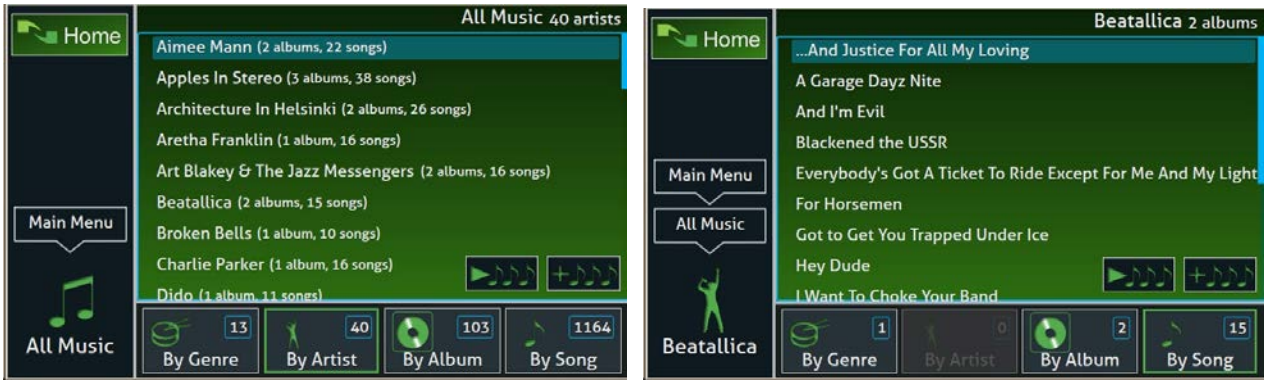

Figure 1. All Music and filtered views. Notice the "breadcrumb" bar at left to orient the user as to where she is in the application hierarchy and to enable instant access to historical states.

Controller paradigm and high-level support for animations. For this reason we chose the AIR runtime and the Flex component model from Adobe. While the initial demo target is a Windows PC, the choice of Flex and AIR will allow us to support, for example, Android tablets and the iPad in the future with minimal rework.

2.3 Feature Set

While we intend to extend the prototype to eventually encompass most of the functional domains in a modern navigation system, for time reasons we are focusing initially on the music domain.

2.3.1 Local Music Access

On startup VoiceCar indexes the contents of attached USB devices and a designated folder on the hard drive, and it builds a relational database of the genres, artists, albums and songs present. The music content can then be accessed in two different ways:

2.3.1.1 Hierarchical Browsing

This style of interaction works well for users who don't know precisely what they want to hear, or who aren't sure whether a particular song or album by a given artist has been copied to the device. In this paradigm a typical voice command sequence might be "Music," then "By Genre," "Jazz," and "John Coltrane," or perhaps the three-step sequence depicted in Figure 1.

2.3.1.2 One-Shot Searching

VoiceCar also allows users to at any time issue one-shot commands like "play X" or "play X by Y," where X and Y correspond to words from any field in the music metadata store.

2.3.2 Remote Music Access

At the same time as the "play X by Y" command is processed by the local, embedded ASR instance, the captured audio is encoded and streamed to a server-based recognizer with a wider vocabulary and a more relaxed language model. When both results come back, if the embedded result has a sufficiently low confidence score attached to it, it is deduced that the command likely referred to music not present in the local collection. The remote result is parsed instead for artist, track, and/or album tags, and the GUI's streaming player is launched.[1]

3. USABILITY AFFORDANCES

VoiceCar accommodates different types of users (novice to familiar to expert) and different contexts of use (e.g., congested urban driving interspersed with occasional stops at traffic signals). It does so by allowing modality cross-over during interactions and by supporting different initiation and pacing strategies for multi-step voice input.

3.1 Modality Cross-over

Envision a harried urban driver who has been carrying out a step-by-step hierarchical browsing interaction using speech, but she is now pulling to a stop at a traffic signal. For subsequent steps in the interaction, she can simply switch to the touchscreen or rotary knob without any need to restart at the root of the hierarchy. Conversely, if a previously uncrowded highway suddenly becomes congested due to construction or an accident, a VoiceCar user who has begun an interaction using the touchscreen can conscientiously put both hands back on the steering wheel, press the push-to-talk (PTT) button, and continue the interaction by voice.

3.2 Dialogs

3.2.1 Background

Mixed-initiative voice dialogs are those in which the pace of the interaction is controlled by the system once the initial PTT button press has been received. Once the user initiates the dialog via PTT press, the system takes control of the pace. Typically it plays a recorded or synthesized prompt to explain what kind of input is expected, then immediately it plays the listening beep and opens the microphone for input. The interpretation of this input may trigger another immediate prompt/beep/response cycle, either for traversal into a deeper node of the menu hierarchy or, for example, to obtain a Yes/No confirmation or a selection from an N-best result list.

System-paced interactions like this are standard in the industry today, and many experienced users prefer them, because

[1] Currently Grooveshark (http://grooveshark.com) streams are supported, but we plan to support other streaming services in the future.

unnecessary PTT button presses are avoided and the overall duration of the multi-step interaction is kept as short as possible. However, those new to speech dialog systems, or those new to a particular dialog and grammar implementation, may have trouble understanding what they can say or when it is their turn to speak. Consequently they may issue an invalid command, or keep silent entirely while they try to think of a valid command. Meanwhile the system will notice their silence and re-prompt them with a phrase like "I didn't get that," which may make the user feel nervous or uneasy.

3.2.2 Kinder, Gentler Dialogs

VoiceCar supports system-paced dialogs because of the efficiency advantages mentioned above, but makes two subtle tweaks to their behavior that we feel improve their suitability for novice users and/or stressful driving situations.

Firstly, the delay between iterations of the prompt/beep/response cycle—between steps in the dialog, effectively—is configurable within the GUI. Expert users who do not need even a quick glance to the screen to confirm their expectations might set this delay between zero and one seconds, whereas speech UI novices or those experiencing the system for the first time might prefer a delay of two or three seconds (or even longer).

Secondly, due to the parallel data models of GUI and VUI, and the seamless modality cross-over described above, users feel less stressed and less rushed to respond "before it's too late." A silence timeout from the speech recognizer does not mean any lost progress; after one or two silences (this is configurable) the system simply plays a "pausing" earcon and closes the microphone. Interaction can then be resumed as soon as driving conditions allow, either via a purely tactile affordance or via PTT plus speech.

3.3 "Anti-Dialogs"

Even with these behavior tweaks, some users might prefer to turn dialog auto-advance off completely, in other words to set the delay between prompt/beep/response cycles to infinity. This results in an interaction paradigm one could call an "anti-dialog," where the user carries out the discrete, individual atoms of an interaction entirely at her own pace, pausing, considering, and perhaps back-tracking before eventually accomplishing the task she set out to accomplish.

Or perhaps—more interestingly—accomplishing a different task than the one she originally had in mind, due to the vagaries of mood or inspiration. Music-domain tasks are particularly susceptible to these sorts of swings of intent. People quite frequently begin browsing their collections with one thing in mind, and then end up choosing something completely different. Going in with a vague desire for Joni Mitchell and coming out convinced you want to hear Regina Spektor is the kind of "happy accident" that couldn't happen with a traditional music dialog (and is probably the only happy accident you'll ever have in a car!).

4. FUTURE WORK

We plan to expand VoiceCar to cover more functional domains, including hybrid (embedded/cloud) Point of Interest search and SMS dictation. Also on the horizon is integration with Nuance's handwriting recognition technology, as we are seeing more cars that incorporate touchpads into their HMI.

Before we get too far with functionality enhancement, however, it would behoove us to conduct one or more usability studies in driving simulators or instrumented vehicles in order to vet our assumptions regarding ease of use and suitability across different user groups and driving scenarios.

5. ACKNOWLEDGMENTS

There are too many contributors to acknowledge everyone by name, but some particularly vital work has been done by Lars König, Andrew Knowles, Rick Megley, Daniel Regert and Slawek Jarosz of Nuance and by Björn Kunz and Sandro Castronovo of DFKI, our GUI partner for this project.

6. REFERENCES

[1] Barón, A. and Green, P. 2006. Safety and Usability of Speech Interfaces for In-Vehicle Tasks while Driving: A Brief Literature Review. Technical Report UMTRI 2006-5. Ann Arbor, MI: University of Michigan Transportation Research Institute.

[2] Castronovo, S., Mahr, A., Pentcheva, M. and Müller, C. 2010. Multimodal dialog in the car: combining speech and turn-and-push dial to control comfort functions. In *INTERSPEECH-2010*, 510-513.

[3] Garay-Vega, L., Pradhan, A., Weinberg, G., Schmidt-Nielsen, B., Harsham, B., Shen, Y., Divekar, G., Romoser, M., Knodler, M. and Fisher, D. 2010. Evaluation of Different Speech and Touch Interfaces to In-vehicle Music Retrieval Systems. *Accident Analysis & Prevention* 42, Issue 3 (May 2010).

[4] Kun, A., Paek, T. and Medenica, Z. 2007. The Effect of Speech Interface Accuracy on Driving Performance. In *Interspeech-2007,* 1326-1329.

[5] Maciej, J. and Vollrath, M. 2009. Comparison of manual vs. speech-based interaction with in-vehicle information system. *Accident Analysis & Prevention* 41, 924-930.

[6] Mattes, S. 2003. The Lane-Change-Task as a tool for driver distraction evaluation. In *Proceedings of IGfA*.